Delay-Independent Stability and Performance of Distributed Congestion Control

Yueping Zhang, *Student Member, IEEE*, Seong-Ryong Kang, *Student Member, IEEE*, and Dmitri Loguinov, *Member, IEEE*

***Abstract*—Recent research efforts to design better Internet transport protocols combined with scalable Active Queue Management (AQM) have led to significant advances in congestion control. One of the hottest topics in this area is the design of discrete congestion control algorithms that are asymptotically stable under heterogeneous feedback delay and whose control equations do not explicitly depend on the RTTs of end-flows. In this paper, we first prove that single-link congestion control methods with a stable *radial* Jacobian remain stable under arbitrary feedback delay (including heterogeneous directional delays) and that the stability condition of such methods does *not* involve any of the delays. We then extend this result to generic networks with *fixed consistent* bottleneck assignments and max–min network feedback. To demonstratethe practicality of the obtained result, we change the original controller in Kelly *et al.*'s work ["Rate Control for communication networks: Shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, no. 3, pp. 237–252, March 1998] to become robust under random feedback delay and fixed constants of the control equation. We call the resulting framework *Max–min Kelly Control* (MKC) and show that it offers smooth sending rate, exponential convergence to efficiency, and fast convergence to fairness, all of which make it appealing for future high-speed networks.**

***Index Terms*—Asymptotic stability, congestion control, heterogeneous delay.**

I. Introduction

OVER the last 15 years, Internet congestion control has evolved from binary-feedback methods of AIMD/TCP [2], [33] to the more exciting developments based on optimization theory [22], [23], game theory [11], [19], and control theory [10], [11], [24], [26]. It is widely recognized that TCP's congestion control in its current shape is inadequate for very high-speed networks and fluctuation-sensitive real-time multimedia. Thus, a significant research effort is currently under way (e.g., [5], [6], [9], [12], [15], [16], [19], [29], and [32]) to better understand the desirable properties of congestion control and develop new algorithms that can be deployed in future AQM (Active Queue Management) networks.

One of the most important factors in the design of congestion control is its *asymptotic stability*, which is the capability of the protocol to avoid oscillations in the steady-state and properly respond to external perturbations caused by the arrival/departure of flows, variation in feedback, and other transient effects. Stability proofs for distributed congestion control become progressively more complicated as feedback delays are taken into account, which is especially true for the case of *heterogeneous* delays where each user i receives its network feedback delayed by a random amount of time D_i. Many existing papers (e.g., [4], [10]–[12], [17], [18], [19], and [23]) model all users with homogeneous delay $D_i = D$ and do not take into account the fact that end users in real networks are rarely (if ever) synchronized. Several recent studies [20], [24], [27] successfully deal with heterogeneous delays; however, they model D_i as a deterministic metric and require that end flows (and sometimes routers) dynamically adapt their equations based on feedback delays, which potentially leads to RTT unfairness, increased overhead, and other side effects (such as probabilistic stability).

In this paper, we set our goal to build a discrete congestion control system that maintains both stability and fairness under heterogeneously delayed feedback, allows users to use fixed parameters of the control equation, and admits a low-overhead implementation inside routers. We solve this problem by showing that any single-link max–min fair system with a stable radial Jacobian remains asymptotically stable under arbitrary directional delays, extend this result to multilink networks under fixed bottleneck assignments, and apply it to the original controller proposed by Kelly *et al.* [15]. We call the result of these efforts *Max–min Kelly Control* (MKC) and demonstrate that its stability and fairness do not depend on any parameters of the network (such as delay, path length, or the routing matrix of end users). We also show that with a proper choice of AQM feedback, MKC converges to efficiency exponentially fast, exhibits stability and fairness under *random* delays, converges to fairness almost as quickly as AIMD, and does not require routers to estimate any parameters of individual flows.

By isolating bottlenecks along each path and responding only to the most-congested resource, the MKC framework allows for very simple stability proofs, which we hope will lead to a better understanding of Kelly's framework in the systems community and eventually result in an actual implementation of these methods in real networks. Our initial thrust in this direction includes `ns2` simulations of MKC, which show that finite time-averaging of flow rates inside each router coupled with a naive implementation of end-user functions leads to undesirable transient oscillations, which become more pronounced when directional delays $D_i^{\rightarrow}$ and $D_i^{\leftarrow}$ to/from each router increase. We overcome this drawback with simple changes at each end user and confirm that the theoretically predicted monotonic convergence of MKC is achievable in real networks, even when the

Manuscript received March 9, 2005; revised May 31, 2006; approved by the IEEE/ACM TRANSACTIONS ON NETWORKING Editor F. Paganini. This work was supported by NSF Grants CCR-0306246, ANI-0312461, CNS-0434940, and CNS-0519442. An earlier version of this paper appeared in ACM SIGCOMM 2004.

The authors are with the Department of Computer Science, Texas A&M University, College Station, TX 77843-77840 USA (e-mail: yueping@cs.tamu.edu; skang@cs.tamu.edu; dmitri@cs.tamu.edu).

Digital Object Identifier 10.1109/TNET.2007.896169

routers do not know the exact combined rate of end-flows at any time instant n. We also show that our algorithms inside the router incur low overhead (which is less than that in XCP [12] or RED [7]) and require only one addition per arriving packet and two variables per router queue.

The rest of this paper is organized as follows. In Section II, we review related work. In Section III, we study delayed stability and steady-state resource allocation of the classic Kelly controls. In Section IV, we present MKC and prove its delay-independent stability. In Section V, we evaluate convergence properties and steady-state properties of MKC. In Section VI, we discuss various implementation issues of MKC and examine its performance under highly variable delays through `ns2` simulations. In Section VII, we conclude our work and suggest directions for future research.

II. Background

A. Delay-Dependent Congestion Control

Recently, a large amount of theoretical and experimental work has been conducted on designing robust congestion controls. One direction is to model the network from an optimization or game-theoretic point of view [11], [17]–[19], [23]. The original work by Kelly *et al.* [14], [15] offers an economic interpretation of the resource–user model, in which the entire system achieves its optimal performance by maximizing the individual utility of each end user. To implement this model in a decentralized network, Kelly *et al.* describe two algorithms (*primal* and *dual*) and prove their global stability in the absence of feedback delay. However, if feedback delay is present in the control loop, stability analysis of Kelly controls is nontrivial and currently forms an active research area [4], [10], [20], [24], [27], [29].

Recall that in Kelly's framework [15], [24], each user $i \in [1, N]$ is given a unique route r_i that consists of one or more network resources (routers). Feedback delays in the network are heterogeneous and directional. The forward and backward delays between user i and resource j are denoted by $D_{ij}^{\rightarrow}$ and $D_{ij}^{\leftarrow}$, respectively. Thus, the round-trip delay of user i is the summation of its forward and backward delays with respect to any router $j \in r_i$: $D_i = D_{ij}^{\rightarrow} + D_{ij}^{\leftarrow}$. Under this framework, Johari *et al.* discretize Kelly's primal algorithm as follows [10]:

$$x_i(n) = x_i(n-1) + \kappa_i(\omega_i - x_i(n-D_i)\eta_i(n)) \quad (1)$$

where κ_i is a strictly positive gain parameter, ω_i can be interpreted as the willingness of user i to pay the price for using the network, and the network feedback

$$\eta_i(n) = \sum_{j \in r_i} p_j(n - D_{ij}^{\leftarrow}) \quad (2)$$

of user i is the aggregate price p_j of all resources j in its path r_i. Here, $p_j(n)$ is a function of the combined rate $y_j(n)$ of all incoming flows at router j:

$$y_j(n) = \sum_{u \in s_j} x_u(n - D_{uj}^{\rightarrow}) \quad (3)$$

where s_j represents the set of users sharing resource j. Note that we use a notation in which $D_i = 1$ means immediate (i.e., most recent) feedback and $D_i \geq 2$ implies delayed feedback.

Next, recall that for a homogeneous delay D, system (1) is locally stable if [10]

$$\kappa_i \sum_{j \in r_i} \left(\left(p_j + p_j' \sum_{u \in s_j} x_u\right)\Bigg|_{x_u^*} \right) < 2 \sin\left(\frac{\pi}{2(2D-1)}\right) \quad (4)$$

where x_u^* is the stationary point of user u and $p_j(\cdot)$ is assumed to be differentiable at x_u^*.

For heterogeneous delays, a combination of conjectures made by Johari *et al.* [10], derivations in Massoulié [24], and the proofs of Vinnicombe [27] suggest that delay D in can be simply replaced with individual delays D_i to form a system of N stability equations; however, the proof exists only for the *continuous* version of (1) and leads to the following sufficient stability condition [27]:

$$\kappa_i \sum_{j \in r_i} \left(\left(p_j + p_j' \sum_{u \in s_j} x_u\right)\Bigg|_{x_u^*} \right) < \frac{\pi}{2D_i}. \quad (5)$$

Inspired by Kelly's optimization framework, one additional method called MaxNet is proposed in [31] and is aimed at improving convergence properties [30] of traditional models of additive feedback. In MaxNet, each user i obtains feedback $\eta_i(t) = \max_{j \in r_i} p_j(t)$ from the most congested router in its path and applies $\eta_i(t)$ to an unspecified control law $x_i(t) = f_i(\eta_i(t))$. Based on the technique developed in [26], the authors prove that MaxNet is locally stable in generic networks with fixed bottleneck assignments if $0 < f_i'(\eta_i^*) < x_i^*/D_i$, where x_i^* and η_i^* are, respectively, the equilibrium rate and stationary feedback of flow i.

B. Delay-Independent Congestion Control

To the best of our knowledge, the first delay-independent stability condition is due to Vinnicombe, who proposes and examines the following continuous fluid model of a network with sources operating TCP-like algorithms [28]:

$$\dot{x}_i(t) = \frac{x_i(t-D_i)}{D_i}(\alpha_i(t) - (\alpha_i(t) + \beta_i(t))\eta_i(t)) \quad (6)$$

where $\alpha_i(t) = a(x_i(t)D_i)^n$, $\beta_i(t) = b(x_i(t)D_i)^m$, a, b, m, n are constants, $\eta_i(t)$ is the network feedback, and link price $p_j(t) = (y_j(t)/C_j)^B$ is an approximation of packet loss at link j of capacity C_j and buffer size B. It is proven in [28] that the above controller is locally stable if

$$a(x_i^* D_i)^n < \frac{1}{B}. \quad (7)$$

Setting $n = 0$ and $a < 1/B$, the resulting system achieves delay-independent stability.

An additional result is available from [34], where Ying *et al.* consider the following variant of controller (6):

$$\dot{x}_i(t) = \kappa_i x_i(t-D_i)\left(\frac{1}{x_i^n(t)} - x_i^m(t)\eta_i(t)\right) \quad (8)$$

where κ_i is a constant. The authors prove that (8) is globally stable regardless of delay in general network topologies if $m + n > B$. This work is similar in spirit to ours; however, the analysis and proposed methods are different.

III. Classic Kelly Control

In this section, we discuss intuitive examples that explain the cryptic formulas in the previous section and demonstrate in simulation how delays affect the stability of Kelly controls (1). We then show that the original Kelly control [15], or any mechanism that relies on the *sum* of feedback functions from individual routers, exhibits a tradeoff between linear convergence to efficiency and persistent stationary packet loss. We subsequently overcome both limitations in Section IV.

A. Delayed Stability Example

The following example illustrates stability problems of (1) when feedback delays are large. We assume a single-source, single-link configuration and utilize a congestion indication function that computes the estimated packet loss using instantaneous arrival rates:

$$p(n) = \frac{x(n) - C}{x(n)} \tag{9}$$

where C is the link capacity and $x(n)$ is the flow rate at discrete step n. We note that the price function $p(n)$ in the original Kelly control is nonnegative; however, as shown in [35], this results in slow linear AIMD-like probing for link capacity until the slowest link in the path is fully utilized, which is generally considered too slow for high-speed networks. Thus, under AQM feedback assumed throughout this paper, we allow *negative* values in (9), which signals the flow to increase its sending rate when $x(n) < C$. In Section V-A, we show that the negative component of packet loss (9) improves convergence to efficiency from linear to exponential.

Applying (9) to Kelly control (1) yields a linear end-flow equation

$$x(n) = x(n-1) + \kappa\omega - \kappa(x(n-D) - C). \tag{10}$$

Next, assume a particular set of parameters: $\kappa = 1/2$, $\omega = 10$ mb/s, and $C = 1000$ mb/s. Solving the condition in (4), we have that the system is stable if and only if delay D is less than four time units. As illustrated in Fig. 1(a), delay $D = 1$ keeps the system stable and monotonically convergent to its stationary point. Under larger delays $D = 2$ and $D = 3$ in Fig. 1(b) and (c), the flow exhibits progressively increasing oscillations before entering the steady state. Eventually, as soon as D becomes equal to four time units, the system diverges as shown in Fig. 1(d).

Using the same parameter κ and reducing ω to 20 kb/s, we examine (10) via `ns2` simulations, in which a single flow passes through a link of capacity 50 mb/s. We run the flow in two network configurations with the round-trip delay equal to 90 and 120 ms, respectively. As seen in Fig. 2, the first flow reaches its steady state after decaying oscillations, while the second flow exhibits no convergence and periodically overshoots capacity C by 200%.

Since Kelly controls are unstable unless condition (4) is satisfied [10], a natural strategy to maintain stability is for each end user i to adaptively adjust its gain parameter $\kappa_i \sim 1/D_i$ such that (4) is not violated. However, this method depends on reliable estimation of round-trip delays D_i and leads to unfairness between the flows with different RTTs.

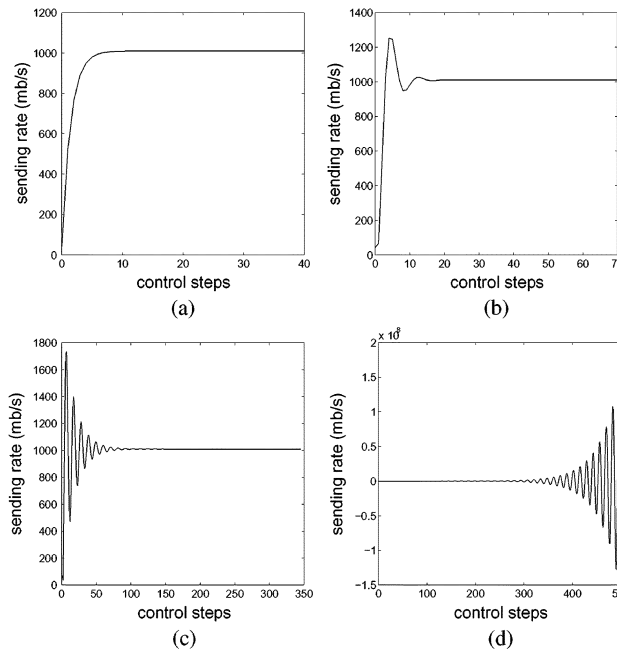


Fig. 1. Stability of Kelly control under different feedback delays ($\kappa = 1/2$, $\omega = 10$ mb/s, and $C = 1000$ mb/s). (a) $D = 1$, (b) $D = 2$, (c) $D = 3$, and (d) $D = 4$.

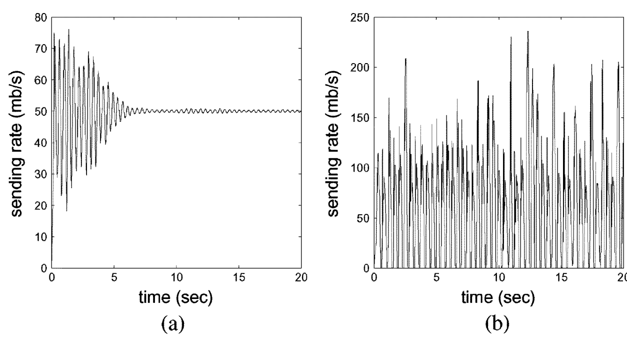


Fig. 2. Simulation results of the classic Kelly control under different delays ($\kappa = 1/2$, $\omega = 20$ kb/s, $C = 50$ mb/s). (a) $D = 90$ ms and (b) $D = 120$.

B. Stationary Rate Allocation

As mentioned in the previous subsection, price function (9) should allow negative values such that the convergence speed of Kelly control is improved from linear to exponential. However, we show next that this modification presents a problem in the stationary resource allocation. Consider a network of M resources and N homogeneous users (i.e., with the same parameters κ and ω). Further assume that resource j has capacity C_j, user i utilizes route r_i of length M_i (i.e., $M_i = |r_i|$), and packet loss $\eta_i(n)$ fed back to user i is the *aggregate* feedback from all resources in path r_i. We further assume that there is no redundancy in the network (i.e., each user sends its packets through at least one resource and all resources are utilized by at least one user). Thus, we can define routing matrix $A_{N\times M}$ such that $A_{ij} = 1$ if user i passes through resource j (i.e., $j \in r_i$) and $A_{ij} = 0$ otherwise. Further denote the jth column of A by vector $\mathbf{V}_j$. Clearly, $\mathbf{V}_j$ identifies the set s_j of flows passing through router j.

Let $\mathbf{x}_j(n) = (x_1(n - D_{1j}^{\rightarrow}), x_2(n - D_{2j}^{\rightarrow}), \ldots, x_N(n - D_{Nj}^{\rightarrow}))$ be the vector of sending rates of individual users observed at router j at time instant n. In the spirit of (9), the packet loss of resource j at instant n can be expressed as

$$p_j(n) = \frac{\mathbf{x}_j(n) \cdot \mathbf{V}_j - C_j}{\mathbf{x}_j(n) \cdot \mathbf{V}_j} \tag{11}$$

where the dot operator represents vector multiplication. Then, we have the following result.

Lemma 1: Let $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_N^*)$ be the stationary rate allocation of Kelly control (1) with packet-loss function (11). Then $\mathbf{x}^*$ satisfies

$$\sum_{i=1}^{N} M_i x_i^* = \sum_{j=1}^{M} C_j + N\omega. \tag{12}$$

Proof: In the steady state, we can write the control equation of user i as follows:

$$\begin{aligned} x_i^* &= (1 - \kappa\eta_i^*)x_i^* + \kappa\omega \\ &= \left(1 - \kappa \sum_{j \in r_i} \frac{\mathbf{x}^* \cdot \mathbf{V}_j - C_j}{\mathbf{x}^* \cdot \mathbf{V}_j}\right) x_i^* + \kappa\omega \end{aligned} \tag{13}$$

where η_i^* denotes the stationary feedback seen by user i. Using simple manipulations in (13), we have

$$M_i x_i^* - \sum_{j \in r_i} \left(\frac{x_i^* C_j}{\mathbf{x}^* \cdot \mathbf{V}_j}\right) = \omega. \tag{14}$$

Taking a summation of (14) for all N users, we get

$$\sum_{i=1}^{N} M_i x_i^* = \sum_{i=1}^{N} \sum_{j \in r_i} \left(\frac{x_i^* C_j}{\mathbf{x}^* \cdot \mathbf{V}_j}\right) + N\omega. \tag{15}$$

Assuming no redundant users or resources, we can rewrite (15) as follows:

$$\begin{aligned} \sum_{i=1}^{N} M_i x_i^* &= \sum_{j=1}^{M} \sum_{i \in s_j} \left(\frac{x_i^* C_j}{\mathbf{x}^* \cdot \mathbf{V}_j}\right) + N\omega \\ &= \sum_{j=1}^{M} (\mathbf{x}^* \cdot \mathbf{V}_j) \left(\frac{C_j}{\mathbf{x}^* \cdot \mathbf{V}_j}\right) + N\omega \\ &= \sum_{i=1}^{M} C_i + N\omega \end{aligned} \tag{16}$$

which completes the proof. ∎

Lemma 1 provides a connection between the stationary resource allocation and the path length of each flow. Note that according to (12), the stationary rates x_i^* are constrained by the capacity of *all* resources instead of by that of individual bottlenecks. In fact, this observation shows an important difference between real network paths, which are limited by the *slowest* resource, and the model of proportional fairness augmented with (11), which takes into account the capacity of *all* resources in the network. As demonstrated in [35], this difference leads to significant overflow of slow routers and underutilization of fast routers along a given path.

In the next section, we propose a new controller that overcomes both drawbacks of controller (1) (i.e., instability under delay and linear convergence to efficiency).

IV. Stable Congestion Control

A. Max–Min Kelly Control

We start our discussion with the following observations. First, we notice that in the classic Kelly control (1), the end user decides its current rate $x_i(n)$ based on the most recent rate $x_i(n-1)$ and delayed feedback $\eta_j(n - D_{ij}^{\leftarrow})$. Since the latter carries information about $x_i(n - D_i)$, which was in effect *RTT time units earlier*, the controller in (1) has no reason to involve $x_i(n-1)$ in its control loop. Thus, the sender quickly becomes unstable as the discrepancy between $x_i(n-1)$ and $x_i(n - D_i)$ increases. One natural remedy to this problem is to retard the reference rate to become $x_i(n - D_i)$ instead of $x_i(n-1)$ and allow the feedback to accurately reflect network conditions with respect to the first term of (1).

Second, to avoid unfairness[1] between flows, one must fix the control parameters of all end users and establish a uniform set of equations that govern the system. Thus, we create a new notation in which $\kappa_i\omega_i = \alpha$, $\kappa_i = \beta$ and discretize Kelly control as follows:

$$x_i(n) = x_i(n - D_i) + \alpha - \beta\eta_i(n)x_i(n - D_i) \tag{17}$$

where $\eta_i(n)$ is the congestion indication function of user i.

Next, to overcome the problems of proportional fairness described in the previous section and utilize negative network feedback, we combine (17) with max–min fairness (this idea is not new [12], [30]), under which the routers only feed back the packet loss of the *most congested* resource instead of the combined packet loss of all links in the path, as follows:

$$\eta_i(n) = \max_{j \in r_i} p_j(n - D_{ij}^{\leftarrow}) \tag{18}$$

where $p_j(\cdot)$ is the congestion indication function of individual routers that depends only on the aggregate arrival rate $y_j(n)$ of end users.

We call the resulting controller (17) MKC and emphasize that flows congested by the same bottleneck receive the *same* feedback and behave independently of the flows congested by the other links (see below for a justification of this). Therefore, in the rest of this paper, we study the single-bottleneck case since each MKC flow is always congested by only *one* router. Implementation details of how routers should feed back function (18) and how end flows track the changes in the most congested resource are presented in the simulation section.

B. Delay-Independent Stability

Before restricting our analysis to MKC, we examine a wide class of delayed control systems, whose stability directly follows from that of the corresponding undelayed systems. We subsequently show that MKC belongs to this category and obtain a very simple proof of its stability. We start by introducing the following definition of induced matrix norms, which is used later in the paper.

[1]While "fairness" is surely a broad term, we assume its max–min version in this paper.

Definition 1 [8]: Matrix norm $\|\cdot\|_\alpha$ is induced by a given vector norm $\|\cdot\|_\alpha$ if

$$\|A\|_\alpha = \sup_{x\neq 0} \frac{\|Ax\|_\alpha}{\|x\|_\alpha}. \tag{19}$$

For instance, the spectral norm $\|A\|_2 = \sqrt{\rho(A^*A)}$ (where A^* is the conjugate transpose of A) is induced by the L^2 vector norm $\|\mathbf{x}\|_2 = (\sum_{i=1}^N |x_i|^2)^{1/2}$. Using this definition, we introduce the class of *radial* matrices as follows.

Definition 2: Matrix A is radial (also called normaloid in the context of operator norms) if $\rho(A) = \|A\|_2$.

It is not difficult to see that radial matrices include symmetric ($a_{ij} = a_{ji}$), skew-symmetric ($a_{ij} = -a_{ji}$), Hermitian ($A = A^*$), skew-Hermitian ($A = -A^*$), unitary ($A^* = A^{-1}$), and circulant matrices (see below for the definition). Then, we have the following theorem.

Theorem 1: Assume an undelayed linear system $\mathcal{L}$ with N flows, as follows:

$$x_i(n) = \sum_{j=1}^N a_{ij} x_j(n-1). \tag{20}$$

If coefficient matrix $A = (a_{ij})$ is radial, then the following system $\mathcal{L}_D$ with arbitrary heterogeneous directional delays $D_i^{\rightarrow}$ and $D_i^{\leftarrow}$:

$$x_i(n) = \sum_{j=1}^N a_{ij} x_j(n - D_j^{\rightarrow} - D_i^{\leftarrow}) \tag{21}$$

is asymptotically stable if and only if $\mathcal{L}$ is stable.

Proof: We start with proving the sufficient condition. Assume that $\mathcal{L}$ is stable, i.e., $\rho(A) < 1$. Applying z-transform to system (21), we obtain

$$\mathbf{H}(z) = Z_2 A Z_1 \mathbf{H}(z) \tag{22}$$

where $Z_1 = \mathrm{diag}(z^{-D_i^{\rightarrow}})$ and $Z_2 = \mathrm{diag}(z^{-D_i^{\leftarrow}})$ are the diagonal matrices of directional delays $D_i^{\rightarrow}$ and $D_i^{\leftarrow}$, and $\mathbf{H}(z)$ is the vector of z-transforms of each flow rate x_i:

$$\mathbf{H}(z) = (H_1(z), H_2(z), \ldots, H_N(z))^T. \tag{23}$$

Notice that system (21) is stable if and only if all poles of its z-transform $\mathbf{H}(z)$ are within the unit circle in the z-plane [13]. To examine this condition, reorganize the terms in (22), as follows:

$$(Z_2 A Z_1 - I)\mathbf{H}(z) = 0. \tag{24}$$

Next, notice that the poles of $\mathbf{H}(z)$ are simply the roots of

$$\det(Z_2 A Z_1 - I) = 0. \tag{25}$$

Thus, ensuring that all roots of (25) are inside the open unit circle will be both sufficient and necessary for system (21) to be stable. Bringing in the notation $\mathcal{F}(z) = \det(Z_2 A Z_1 - I)$, we can rewrite $\mathcal{F}(z)$ as follows:

$$\begin{aligned}\mathcal{F}(\ddagger) &= \det\left(Z_2\left[A - Z_2^{-1} I Z_1^{-1}\right] Z_1\right)\\ &= \det(Z_2)\det\left(A - Z_2^{-1} Z_1^{-1}\right)\det(Z_1).\end{aligned} \tag{26}$$

Noticing that $\det(Z_1)$ and $\det(Z_2)$ are strictly nonzero for nontrivial z, we can reduce (25) to

$$\mathcal{F}(z) = \det(A - Q(z)) = 0 \tag{27}$$

where $Q(z) = \mathrm{diag}(z^{D_i})$ and $D_i = D_i^{\rightarrow} + D_i^{\leftarrow}$.

To prove that all roots of (27) lie in the open unit circle, we suppose in contradiction that there exists a root $|z_0| \geq 1$ such that $\mathcal{F}(z_0) = 0$. Denote by B matrix $Q(z_0)$. Following [25] and using basic matrix algebra, notice that there exists a nonzero vector v such that $Av = Bv$. Since matrix A is radial and invoking Definitions 1 and 2, we can write $\|A\|_2 = \rho(A) < 1$ and

$$\|A\|_2 = \sup_{x\neq 0}\frac{\|Ax\|_2}{\|x\|_2} \geq \frac{\|Av\|_2}{\|v\|_2} = \frac{\|Bv\|_2}{\|v\|_2}. \tag{28}$$

Since B is diagonal with $|b_{ii}| = |z_0|^{D_i} \geq 1$, Bv is simply a vector $(v_1 b_{11}, \ldots, v_N b_{NN})^T$. This leads to

$$\frac{\|Bv\|_2}{\|v\|_2} = \frac{\left(\sum_{i=1}^N |v_i|^2 |b_{ii}|^2\right)^{1/2}}{\left(\sum_{i=1}^N |v_i|^2\right)^{1/2}} \geq \frac{\left(\sum_{i=1}^N |v_i|^2\right)^{1/2}}{\left(\sum_{i=1}^N |v_i|^2\right)^{1/2}} = 1. \tag{29}$$

Thus, we get that both $\|A\|_2 \geq 1$ and $\|A\|_2 < 1$ must be satisfied simultaneously, which is a contradiction. This means that no $|z_0| \geq 1$ can be a root of $\mathcal{F}(z)$ and that any heterogenous system with a radial stable matrix A is stable under arbitrary delay. Proof of the necessary condition is obvious and omitted for brevity. ∎

Theorem 1 opens an avenue for inferring stability of delayed linear systems based on the coefficient matrices of the corresponding undelayed systems. Moreover, it is easy to see that Theorem 1 applies to nonlinear systems as stated in the following corollary.

Corollary 1: Assume an undelayed N-dimensional nonlinear system $\mathcal{N}$:

$$x_i(n) = f_i(x_1(n-1), \ldots, x_N(n-1)) \tag{30}$$

where $\{f_i | f_i : \mathbb{R}^N \to \mathbb{R}\}$ is the set of nonlinear functions defining the system. If the Jacobian matrix J of this system is radial, system $\mathcal{N}_D$ with arbitrary directional delays $D_i^{\rightarrow}$ and $D_i^{\leftarrow}$

$$x_i(n) = f_i(x_1(n - D_1^{\rightarrow} - D_i^{\leftarrow}), \ldots, x_N(n - D_N^{\rightarrow} - D_i^{\leftarrow})) \tag{31}$$

is locally asymptotically stable in the stationary point $\mathbf{x}^*$ if and only if $\mathcal{N}$ is stable in $\mathbf{x}^*$.

Based on the above principles, we next prove local stability of MKC under heterogeneous feedback delays.

C. Single-Link Stability of MKC

Consider an MKC system with a generic feedback function $\eta_i(n)$ in the form of (18), which we assume is differentiable in the stationary point and has the same first-order partial derivative for all end users. Our goal is to derive sufficient and necessary conditions for the stability of (17) and (18) under arbitrarily delayed feedback.

We first prove MKC's stability in a single-link network containing N users $\{x_1, \ldots, x_N\}$ with corresponding delays to/from the bottleneck router given by $D_i^{\rightarrow}$ and $D_i^{\leftarrow}$. Then, we can simplify (17)–(18) by dropping index j of the bottleneck resource and expanding $\eta_i(n)$ in (17), as follows:

$$x_i(n) = x_i(n - D_i) + \alpha - \beta p(n - D_i^{\leftarrow})x_i(n - D_i) \quad (32)$$

where

$$p(n) = p\left(\sum_{u=1}^{N} x_u(n - D_u^{\rightarrow})\right) \quad (33)$$

is the packet-loss function of the bottleneck router. Notice that $x_i(n - D_i)$ in (32) can be represented as $x_i(n - D_i^{\rightarrow} - D_i^{\leftarrow})$ and that controller (32)–(33) has the same shape as that in (31).

To invoke Theorem 1, our first step is to show stability of the following undelayed version of (32) and (33):

$$\begin{cases} x_i(n) = (1 - \beta p(n-1))x_i(n-1) + \alpha \\ p(n) = p\left(\sum_{u=1}^{N} x_u(n)\right). \end{cases} \quad (34)$$

Theorem 2: Undelayed N-dimensional system (34) with feedback $p(n)$ that is common to all users has a symmetric Jacobian and is locally asymptotically stable if and only if

$$0 < \beta p^* < 2 \quad (35)$$

$$0 < \beta p^* + \beta N x^* \left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*} < 2 \quad (36)$$

where x^* is the fixed point of each individual user, vector $\mathbf{x}^* = (x^*, x^*, \ldots, x^*)$ is the fixed point of the entire system, and p^* is the steady-state packet loss.

Proof: We first derive the stationary point x^* of each individual user. Since all end users receive the same feedback and activate the same response to it, all flows share the bottleneck resource fairly in the steady state, i.e., $x_i(n) = x^*$ for all i. Using simple manipulations in (34), we get the stationary individual rate x^* as follows:

$$x^* = \frac{\alpha}{\beta p^*}. \quad (37)$$

Linearizing the system in $\mathbf{x}^*$:

$$\left.\frac{\partial f_i}{\partial x_i}\right|_{\mathbf{x}^*} = \left.\left(1 - \beta p - \beta x_i \frac{\partial p}{\partial x_i}\right)\right|_{\mathbf{x}^*} \quad (38)$$

$$\left.\frac{\partial f_i}{\partial x_k}\right|_{\mathbf{x}^*} = \left.\left(-\beta x_i \frac{\partial p}{\partial x_k}\right)\right|_{\mathbf{x}^*}, \quad k \neq i \quad (39)$$

where $f_i(\mathbf{x}) = (1 - \beta p(\mathbf{x}))x_i + \alpha$. Since packet loss depends on the *aggregate* rate of all users, $p(n)$ has the same first partial derivative evaluated in the fixed point for all users, which implies that for any users i and k, we have

$$\left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*} = \left.\frac{\partial p}{\partial x_k}\right|_{\mathbf{x}^*}. \quad (40)$$

This observation leads to a simple Jacobian matrix for MKC:

$$J = \begin{pmatrix} a & b & \cdots & b \\ b & a & \cdots & b \\ \vdots & \vdots & \ddots & \vdots \\ b & b & \cdots & a \end{pmatrix} \quad (41)$$

where

$$a = 1 - \beta p^* - \beta x^* \left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*}, \quad b = -\beta x^* \left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*}. \quad (42)$$

Clearly, Jacobian matrix J is circulant[2] and thus its kth eigenvalue λ_k is given by [1]

$$\lambda_k = a + b(\zeta_k + \zeta_k^2 + \zeta_k^3 + \cdots + \zeta_k^{N-1}) \quad (43)$$

where $\zeta_k = e^{i2\pi k/N}$ $(k = 0, 1, \ldots, N-1)$ is one of the Nth roots of unity. We only consider the case of $N \geq 2$; otherwise, the only eigenvalue is simply a. Then, it is not difficult to get the following result:

$$\lambda_k = \begin{cases} a + (N-1)b & \zeta_k = 1 \\ a + b\frac{\zeta_k - \zeta_k^N}{1 - \zeta_k} = a - b & \zeta_k \neq 1 \end{cases} \quad (44)$$

where the last transition holds since $\zeta_k^N = 1$ for all k.

Next, recall that nonlinear system (34) is locally stable if and only if all eigenvalues of its Jacobian matrix J are within the unit circle [13]. Therefore, we get the following necessary and sufficient local stability conditions:

$$\begin{cases} |a - b| < 1 \\ |a + (N-1)b| < 1. \end{cases} \quad (45)$$

To ensure that each λ_i lies in the unit circle, we examine the two conditions in (45) separately. First, notice that $|a - b| = |1 - \beta p^*|$, which immediately leads to the following:

$$0 < \beta p^* < 2. \quad (46)$$

Applying the same substitution to the second inequality in (45), we obtain

$$0 < \beta p^* + \beta N x^* \left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*} < 2. \quad (47)$$

Thus, system (34) is locally stable if and only if both (46) and (47) are satisfied. ■

According to the proof of Theorem 2, Jacobian J of the undelayed system (34) is symmetric and therefore is radial. Combining this property with Corollary 1, we arrive at the following result.

Corollary 2: Heterogeneously delayed MKC (32)–(33) is locally asymptotically stable if and only if (35) and (36) are satisfied.

Corollary 2 is a generic result that is applicable to MKC (17) with a wide class of congestion-indicator functions $\eta_i(n)$. Further note that for a given bottleneck resource with pricing function $p(n)$ and the set of its users, conditions (35) and (36) are easy to verify and do *not* depend on feedback delays, the number of hops in each path, or the routing matrix of all users. This is in contrast to many current studies [10], [24], [27], [29], whose results are dependent on individual feedback delays D_i and the topology of the network.

D. Multi-Link Stability of MKC

As discussed in [36], rigorous global stability analysis of max–min systems under general conditions (e.g., dynamically changing bottlenecks) is a very complicated issue. We leave this problem for future work and instead show MKC's stability in networks with certain types of fixed bottleneck assignments. Specifically, for a given bottleneck assignment $b_1, \ldots, b_N$, where b_i is the link from which flow i receives its feedback, let $x_i \rightarrow x_k$ denote the fact that flow i passes through flow k's

[2]A matrix is called *circulant* if it is square and each of its rows can be obtained by shifting (with wrap-around) the previous row one column right [1].

bottleneck b_k while being restricted at some other link $b_i \neq b_k$. In this case, we call flows x_i and x_k, respectively, *unresponsive* and *responsive* with respect to link b_k. Then, we can construct a directed dependency graph G based on relationship $\rightarrow$ and prove the following property of G under max–min feedback.

Lemma 2: For any system with max–min feedback (18) that can stabilize its bottleneck assignment $b_1, \ldots, b_N$, the resulting dependency graph is acyclic.

Proof: Suppose that the bottleneck assignment does not change over time and the dependency graph has a directed cycle $x_{i_1} \rightarrow \cdots \rightarrow x_{i_k} \rightarrow x_{i_1}$ for some $k \geq 2$. Notice that since flow x_{i_1} is unresponsive with respect to flow x_{i_2}, its stationary packet loss $p^*_{i_1}$ must be larger than $p^*_{i_2}$ (otherwise, x_{i_1} would have switched its bottleneck to b_{i_2}). Generalizing this to the entire cycle, we immediately get a contradiction $p^*_{i_1} > p^*_{i_2} > \cdots > p^*_{i_k} > p^*_{i_1}$. Assuming a consistent tie-breaking rule obeyed by all flows, the above argument applies to cases where multiple routers have equal steady-state loss. ■

Generalizing this lemma, we define a bottleneck assignment as *consistent* if it has an acyclic dependency graph. Then, we have the following result.

Lemma 3: MKC with a consistent bottleneck assignment $b_1, \ldots, b_N$ contains at least one link that has no unresponsive flows.

Proof: Assume in contradiction that each router l has some unresponsive flow u_l passing through it and that this situation persists over time. Take the first unresponsive flow u_1 and notice that it is affected by some other unresponsive flow, which we label u_2, passing through u_1's bottleneck b_{u_1}. This leads to $u_1 \leftarrow u_2$. Repeating this reasoning for u_2, we get $u_1 \leftarrow u_2 \leftarrow u_3$, for some unresponsive flow u_3 at bottleneck b_{u_2}. This process continues and creates an infinite sequence $u_1 \leftarrow u_2 \leftarrow u_3 \leftarrow \ldots$. Since the number of unresponsive flows is finite, there is a point k when the sequence repeats itself (i.e., $u_k = u_j, j < k$), and we obtain a cycle in the dependency graph. ■

Equipped with Lemmas 2 and 3, we next prove MKC's stability under any time-invariant bottleneck assignment.

Theorem 3: Under any stationary bottleneck assignment with feedback (18), MKC is locally asymptotically stable regardless of delay if and only if individual bottlenecks are.

Proof: Since bottlenecks do not shift and MKC relies on max–min feedback, Lemma 2 implies that the dependency graph is acyclic and bottleneck assignment is consistent. Using Lemma 3, there exists at least one router l_1 with no unresponsive flows. Then, it follows that all flows passing through l_1 are bottlenecked by l_1 and their stability is independent of the dynamics of the remaining flows. After the users bottlenecked by l_1 converge to their stationary rates, we can remove l_1 and all of its (constant-rate) flows from the system. The new network still exhibits max–min bottleneck assignment and thus contains some router l_2 that has no unresponsive flows. Repeating this argument for all routers $l_1, \ldots, l_M$, we obtain that the local dynamics of the entire system can be viewed as a system of linear block-diagonal equations with matrix $A = \mathrm{diag}(A_1, \ldots, A_M)$, where $A_l \in \mathbb{R}^{N_l \times N_l}$ is the Jacobian matrix of N_l flows bottlenecked at router l ($\sum_{l=1}^{M} N_l = N$). We conclude that the entire system achieves delay-independent stability if and only if the individual bottlenecks do. ■

E. Exponential MKC

To understand the practical implications of the derivations above, consider a particular packet-loss function $p(n)$ in (33):

$$p(n) = \frac{\sum_{u=1}^{N} x_u(n - D_u^{\rightarrow}) - C}{\sum_{u=1}^{N} x_u(n - D_u^{\rightarrow})} \tag{48}$$

where we again assume a network with a single link of capacity C and N users. This is a rather standard packet-loss function with the exception that we allow it to become negative when the link is underutilized. As we show in the next section, (48) achieves exponential convergence to efficiency, which explains why we call the combination of (32) and (48) *Exponential MKC* (EMKC).

Theorem 4: Heterogeneously delayed single-link EMKC (32), (48) is locally asymptotically stable if and only if $0 < \beta < 2$.

Proof: We first derive the fixed point of EMKC. Notice that in the proof of Theorem 2, we established the existence of a unique stationary point $x^*_i = x^*$ for each flow. Then assuming EMKC packet-loss function (48), we have

$$p^* = \frac{Nx^* - C}{Nx^*}. \tag{49}$$

Combining (49) and (37), we get the stationary point x^* of each end user:

$$x^* = \frac{C}{N} + \frac{\alpha}{\beta}. \tag{50}$$

Denoting by $X(n) = \sum_{i=1}^{N} x_i(n)$ the combined rate of all N end users at time n, the corresponding combined stationary rate X^* is:

$$X^* = Nx^* = C + N\frac{\alpha}{\beta}. \tag{51}$$

Next, recall from Theorem 2 that stability conditions (35)–(36) must hold for the delayed system to be stable. Consequently, we substitute pricing function (48) into (36) and obtain with the help of (51):

$$\beta p^* + \beta N x^* \left.\frac{\partial p(n)}{\partial x(n)}\right|_{\mathbf{x}^*} = \beta p^* + \frac{\beta N x^* C}{N^2 x^{*2}} = \beta. \tag{52}$$

Thus, condition (36) becomes

$$0 < \beta < 2. \tag{53}$$

Notice that in the steady state, packet-loss probability p^* is no larger than one. Hence, the last condition is more conservative than (35), which allows us to conclude that when $0 < \beta < 2$, all eigenvalues of Jacobian matrix J are inside the unit circle. Applying Corollary 2, heterogeneously delayed EMKC defined by (32) and (48) is also locally asymptotically stable if and only if $0 < \beta < 2$. ■

EMKC's multilink stability follows from Theorem 3.

Corollary 3: Under any stationary bottleneck assignment with feedback (18) and (48), EMKC is locally asymptotically stable regardless of delay if and only if $0 < \beta < 2$.

We additionally note that global asymptotic stability of EMKC under homogeneous delay has been proved in [37] and simulations suggest that EMKC is also globally stable under heterogeneous (or even random) delays. All these results support our earlier conclusion that EMKC is a stable and fair controller regardless of delays, which is a requirement for any practical method in the current Internet.

V. Performance of EMKC

A. Convergence to Efficiency

In this section, we show that EMKC converges to efficiency exponentially fast.

Lemma 4: For $0 < \beta < 2$ and constant delay D, the combined rate $X(n)$ of EMKC is globally asymptotically stable and converges to $X^* = C + N\alpha/\beta$ at an exponential rate.

Proof: Since delays do not affect the stability of EMKC, assume a constant feedback delay D and rewrite (32), as follows:

$$x_i(n) = (1 - \beta p(n-D))x_i(n-D) + \alpha \tag{54}$$

where $p(n)$ is the undelayed version of (48). Taking the summation of (54) for all N flows, we get that EMKC's combined rate $X(n) = \sum_{i=1}^{N} x_i(n)$ forms a linear system:

$$\begin{aligned} X(n) &= \left(1 - \beta\frac{X(n-D) - C}{X(n-D)}\right) X(n-D) + N\alpha \\ &= (1-\beta)X(n-D) + \beta C + N\alpha. \end{aligned} \tag{55}$$

It is clear that the above linear system is stable if and only if $0 < \beta < 2$. Since convergence of linear systems implies global asymptotic stability, we conclude that $X(n)$ is globally stable regardless of individual flow trajectories $x_i(n)$.

We next show the convergence speed of $X(n)$. Recursively expanding the last equation, we have:

$$X(n) = (1-\beta)^{\frac{n}{D}}(X_0 - X^*) + X^* \tag{56}$$

where X_0 is the initial combined rate of all flows and $X^* = C + N\alpha/\beta$ is the combined stationary rate. Notice that for $0 < \beta < 2$, the first term in (56) approaches zero exponentially fast and $X(n)$ indeed converges to X^*. ■

From (56), notice that the value of β affects the convergence behavior of EMKC. Specifically, for $0 < \beta \le 1$, the system monotonically converges to the stationary point; however, for $1 < \beta < 2$, the system experiences decaying oscillations before reaching the stationary point, which are caused by the oscillating term $(1-\beta)^{n/D}$ in (56). Thus, in practical settings, β should be chosen in the interval $(0, 1]$, where values closer to 1 result in faster convergence to efficiency.

B. Convergence to Fairness

We next investigate the convergence rate of EMKC to fairness. To better understand how many steps EMKC requires to reach a certain level of max–min fairness, we utilize a simple metric that we call *ε-fairness*. For a given small positive constant ε, a rate allocation $(x_1, x_2, \ldots, x_N)$ is ε-fair, if

$$f = \frac{\min_{i=1}^{N} x_i}{\max_{j=1}^{N} x_j} \ge 1 - \varepsilon. \tag{57}$$

Generally, ε-fairness assesses max–min fairness by measuring the worst-case ratio between the rates of any pair of flows. Given the definition in (57), we have the following result.

Theorem 5: Consider an EMKC network with N users and a bottleneck link of capacity C. Assuming that the system is started in the maximally unfair state, ε-fairness is reached in θ_M steps, where

$$\theta_M = \frac{\left(C + N\frac{\alpha}{\beta}\right)(\log N - \log\varepsilon)}{N\alpha} + \Theta\left(\frac{N\alpha}{C}\right). \tag{58}$$

Proof: Let (x, y) be the pair of initially maximally unfair flows, i.e., the difference between their initial sending rates $\Delta(0) = y(0) - x(0)$, where $y(0) > x(0)$, is maximal among that of any two flows. Notice that under MKC and the assumption of synchronization, (x, y) are always maximally unfair during the entire process till certain fairness threshold is reached. Then we have

$$(i) - x(i) \approx (y(0) - x(0))\left(1 - \frac{N\alpha}{X^*}\right)^i = \Delta(0)\left(1 - \frac{N\alpha}{X^*}\right)^i. \tag{59}$$

Thus, the fairness index at step i becomes

$$\begin{aligned} f(i) = \frac{x(i)}{y(i)} = \frac{y(i) - \Delta(i)}{y(i)} &= 1 - \frac{\Delta(0)\left(1 - \frac{N\alpha}{X^*}\right)^i}{y(i)} \\ &\ge 1 - \frac{\Delta(0)\left(1 - \frac{N\alpha}{X^*}\right)^i}{y^*} \end{aligned} \tag{60}$$

since $y(i) \ge y^*$. Hence, to achieve ε-fairness, we have

$$f(n) \ge 1 - \frac{\Delta(0)\left(1 - \frac{N\alpha}{X^*}\right)^n}{y^*} \ge 1 - \varepsilon \tag{61}$$

which yields

$$\begin{aligned} \theta_M &\le \log_{1 - \frac{N\alpha}{X^*}} \frac{y^*\varepsilon}{\Delta(0)} = \frac{\log\bigl(y^*\varepsilon/\Delta(0)\bigr)}{\log(1 - \frac{N\alpha}{X^*})} \\ &\approx -\frac{X^*\log(y^*\varepsilon/\Delta_0)}{N\alpha} + \Theta\left(\frac{N\alpha}{C}\right). \end{aligned} \tag{62}$$

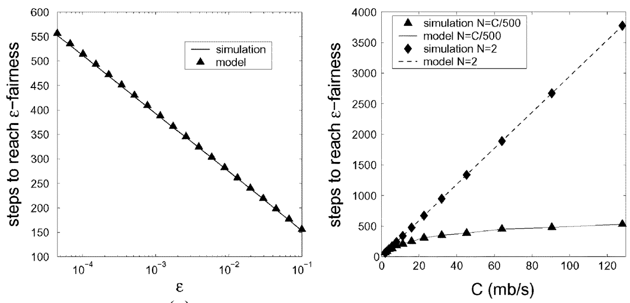
Assuming $N\alpha/C \ll 1$ and substituting $y^* = C/N + \alpha/\beta$ and $X^* = C + N\alpha/\beta$ in (62), we get

$$\begin{aligned} \theta_M &\le -\frac{\left(C + N\frac{\alpha}{\beta}\right)\left(\log\left(\frac{C}{N} + \frac{\alpha}{\beta}\right) + \log\varepsilon - \log\Delta(0)\right)}{N\alpha} \\ &\approx \frac{\left(C + N\frac{\alpha}{\beta}\right)(\log N + \log\Delta(0) - \log C - \log\varepsilon)}{N\alpha} \\ &= \frac{\left(C + N\frac{\alpha}{\beta}\right)(\log N - \log\varepsilon)}{N\alpha}. \end{aligned} \tag{63}$$

Adding the omitted terms on the order of $\Theta(N\alpha/C)$ to (63), we arrive at (58). ■

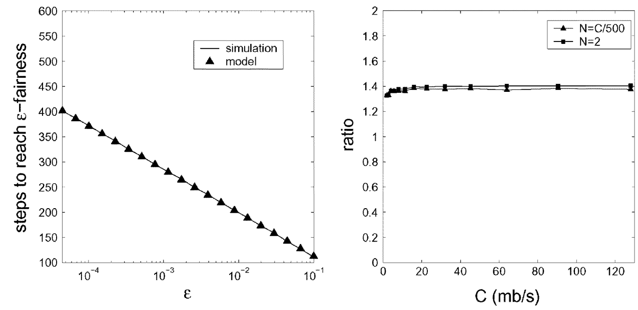
A comparison of model (58) to simulation results is shown in Fig. 3(a) (note that in the figure, the model is drawn as a solid line and simulation results are plotted as isolated triangles). In this example, we use a bottleneck link of capacity $C = 1$ mb/s shared by two EMKC flows, which are initially separated by the maximum distance, i.e., $x_1(0) = 0, x_2(0) = C$. As seen from the figure, the number of steps predicted by (58) agrees with simulation results for a wide range of ε.

As noted in the previous section, parameter β is responsible for the convergence speed to efficiency; however, as seen in (58), it has little effect on the convergence rate to fairness (since typically $N\alpha \ll C$). In contrast, parameter α has no effect on



(a) (b)

Fig. 3. (a) Verification of model (58) against EMKC simulations ($C = 1$ mb/s, $\alpha = 10$ kb/s, and $\beta = 0.5$). (b) Exponential and linear rates of convergence to fairness for EMKC ($\varepsilon = 0.1$).



(a) (b)

Fig. 4. (a) Verification of model (65) against AIMD simulations ($C = 1$ mb/s, $\alpha = 10$ kb/s, and $\beta = 0.5$). (b) Ratio θ_M/θ_A for fixed and variable N.

convergence to efficiency in (56), but instead determines the convergence rate to fairness in the denominator of (58). Also observe the following interesting fact about (58) and the suitability of EMKC for high-speed networks. As C increases, the behavior of θ_M changes depending on whether N remains fixed or not. For a constant N, (58) scales linearly with C; however, if the network provider increases the number of flows as a function of C and keeps $N = \Theta(C)$, ε-fairness is reached in $\Theta(\log C)$ steps. This implies exponential convergence to fairness and very good scaling properties of EMKC in future high-speed networks. Both types of convergence are demonstrated in Fig. 3(b) for constant $N = 2$ and variable $N = \lceil C/500 \rceil$ (for the latter case, C is taken to be in kb/s). As the figure shows, both linear and logarithmic models obtained from (58) match simulations well.

We next compare EMKC's convergence speed to that of rate-based AIMD. Recall that rate-based AIMD (α, β) adjusts its sending rate according to the following rules assuming $\alpha > 0$ and $0 < \beta < 1$:

$$x(t) = \begin{cases} x(t - RTT) + \alpha & \text{per RTT} \\ (1-\beta)x(t - RTT) & \text{per loss.} \end{cases} \tag{64}$$

Theorem 6: Under the assumptions of Theorem 5, rate-based AIMD reaches ε-fairness in θ_A steps, where

$$\theta_A = \frac{\left(C + N\frac{\alpha}{\beta}\right)(\log N - \log \varepsilon)}{-N\alpha \log(1-\beta)/\beta} + \Theta\left(\frac{N\alpha}{C}\right). \tag{65}$$

Proof: Assume that flows are synchronized and reach full link utilization at time instants $\tau_1, \tau_2, \ldots$. We again assume that $N\alpha \ll C$ and neglect the random amount of overshoot, which generally fluctuates between 0 and $N\alpha$. Analysis below focuses on two maximally unfair flows x and y (i.e., $x(0) = 0, y(0) = C$) since these flows solely determine max-min fairness of the system. After packet loss is detected at time τ_j, the immediate rate reduction brings rates $x(\tau_j)$ and $y(\tau_j)$ to $(1-\beta)x(\tau_j)$ and $(1-\beta)y(\tau_j)$ and the combined rate of all users drops to $(1-\beta)C$. Following this reduction, the combined rate is then incremented by $N\alpha$ per RTT until it reaches C at time τ_{j+1}. This implies that at the end of interval $[\tau_j, \tau_{j+1}]$, each flow's rate is increased by $w = \beta C/N$, meaning that flows x and y climb back to $(1-\beta)x(\tau_j) + w$ and $(1-\beta)y(\tau_j) + w$, respectively. Hence, the new rates when the flows hit the efficiency line for the jth time are

$$x(\tau_j) = (1-\beta)x(\tau_{j-1}) + \frac{\beta C}{N} \tag{66}$$

$$y(\tau_j) = (1-\beta)y(\tau_{j-1}) + \frac{\beta C}{N}. \tag{67}$$

It is not difficult to see that the distance between any two flows shrinks exponentially, as follows:

$$\begin{aligned} \Delta(\tau_j) &= y(\tau_j) - x(\tau_j) = (1-\beta)(y(\tau_{j-1}) - x(\tau_{j-1})) \\ &= (1-\beta)\Delta(\tau_{j-1}) = \Delta(0)(1-\beta)^j. \end{aligned} \tag{68}$$

Using simple manipulations, we have max–min fairness

$$\begin{aligned} f(\tau_j) &= \frac{x(\tau_j)}{y(\tau_j)} = \frac{y(\tau_j) - \Delta(\tau_j)}{y(\tau_j)} = 1 - \frac{\Delta(\tau_j)}{y(\tau_j)} \\ &\geq 1 - \frac{(1-\beta)^j \Delta(0)}{C/N} = 1 - q(1-\beta)^j \end{aligned} \tag{69}$$

where constant $q = N\Delta(0)/C$.

The number of packet-loss intervals to reach ε-fairness is no more than $\log_{1-\beta}(\varepsilon/q)$, while the number of increase steps during each packet-loss interval is $\beta C/(N\alpha)$. Thus, the total number of steps to convergence is

$$\begin{aligned} \theta_A &= \left(\left\lceil \frac{\beta C}{N\alpha} \right\rceil\right) \log_{1-\beta} \frac{\varepsilon}{q} \\ &\approx \left(\frac{\beta C}{N\alpha} + 1\right) \log_{1-\beta} \frac{C\varepsilon}{N\Delta(0)} \\ &= \frac{(C + N\alpha/\beta)(\log N + \log \Delta(0) - \log C - \log \varepsilon)}{-N\alpha \log(1-\beta)/\beta} \\ &= \frac{\left(C + N\frac{\alpha}{\beta}\right)(\log N - \log \varepsilon)}{-N\alpha \log(1-\beta)/\beta}. \end{aligned} \tag{70}$$

Accounting for random overshoot and neglected terms, we get (65). ■

Fig. 4(a) verifies that model (65) is also very accurate for a range of different ε. Notice from (58) and (65) that the speed of convergence to fairness between AIMD and EMKC differs by a certain constant coefficient. The following corollary formalizes this observation.

Corollary 4: For the same parameters N, α, β such that $N\alpha \ll C$, AIMD reaches ε-fairness $\theta_M/\theta_A = -\log(1-\beta)/\beta$ times faster than EMKC.

For TCP and $\beta = 0.5$, this difference is by a factor of $2\log 2 \approx 1.39$, which holds regardless of whether N is fixed or not as demonstrated in Fig. 4(b). We should finally note that as term $\Theta(N\alpha/C)$ becomes large, MKC's performance improves and converges to that of AIMD.

C. Packet Loss

As seen in previous sections, EMKC converges to the combined stationary point $X^* = C + N\alpha/\beta$, which is above capacity C. This leads to constant (albeit usually small) packet loss in the steady state. However, the advantage of this framework is that EMKC does not oscillate or react to individual packet losses, but instead adjusts its rate in response to a *gradual* increase in $p(n)$. Thus, a small amount of FEC can provide a smooth channel to fluctuation-sensitive applications such as video telephony and various types of real-time streaming. Besides being a stable framework, EMKC is also expected to work well in wireless networks where congestion-unrelated losses will not cause sudden reductions in the flow rates.

Also notice that EMKC's steady-state packet loss $p^* = N\alpha/(C\beta + N\alpha)$ increases linearly with the number of competing flows, which causes problems in scalability to a large number of flows. However, it still outperforms AIMD, whose increase in packet loss is quadratic as a function of N [21]. Furthermore, if the network provider keeps $N = \Theta(C)$, EMKC achieves *constant* packet loss in addition to exponential convergence to fairness.

Finally, observe that if the router is able to count the number of flows, zero packet loss can be obtained by adding a constant $\Delta = N\alpha/(\beta C)$ to the congestion indication function [3]. However, this method is impractical, since it needs nonscalable estimation of the number of flows N inside each router. Hence, it is desirable for the router to adaptively tune $p(n)$ so that the system is free from packet loss. One such method is AVQ (Adaptive Virtual Queue) proposed in [17] and [20]. We leave the analysis of this approach under heterogeneous delays and further improvements of EMKC for future work.

VI. IMPLEMENTATION

We next examine how to implement scalable AQM functions inside routers to provide proper feedback to MKC flows. This is a nontrivial design issue since the ideal packet loss in (48) relies on the sum of *instantaneous* rates $x_i(n)$, which are never known to the router. In such cases, a common approach is to approximate model (48) with some time-average function computed inside the router. However, as mentioned in the introduction, this does not directly lead to an oscillation-free framework since directional delays of real networks introduce various inconsistencies in the feedback loop and mislead the router to produce incorrect estimates of $X(n) = \sum_i x_i(n)$.

In what follows in this section, we provide a detailed description of various AQM implementation issues and simulate EMKC in `ns2` under heterogeneous (including time-varying) feedback delays.

A. Packet Header

As shown in Fig. 5, the MKC packet header consists of two parts—a 16-byte *router* header and a 4-byte *user* header. The router header encapsulates information that is necessary for the router to generate precise AQM feedback and subsequently for the end user to adjust its sending rate. The id field is a unique label that identifies the router that generated the feedback (e.g., its IP address). This field is used by the flows to detect changes in bottlenecks, in which case they wait for an extra RTT before responding to congestion signals of the new router. The seq field is a local variable incremented by the router each time it produces a new value of packet loss p (see below for more). Finally, the Δ field carries the length of the averaging interval used by the router in its computation of feedback.

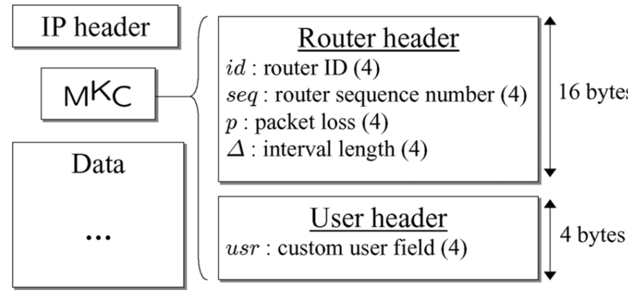


Fig. 5. Packet format of MKC.

The usr field is necessary for end flows to determine the rate $x_i(n - D_i)$ that was in effect RTT time units earlier. The simplest way to implement this functionality is to inject the value of $x_i(n)$ into each outgoing packet and then ask the receiver to return this field in its acknowledgments. A slightly more sophisticated usage of this field is discussed later in this section.

B. The Router

Recall that MKC decouples the operations of users and routers, allowing for a scalable decentralized implementation. The major task of the router is to generate its AQM feedback and insert it in the headers of all passing packets. However, notice that the router never knows the exact combined rate of incoming flows. Thus, to approximate the ideal computation of packet loss, the router conducts its calculation of $p(n)$ on a discrete time scale of Δ time units. For each packet arriving within the *current* interval Δ, the router inserts in the packet header the feedback information computed during the *previous* interval Δ. As a consequence, the feedback is retarded by Δ time units inside the router in addition to any backward directional delays $D_i^{\leftarrow}$. Since MKC is robust to feedback delay, this extra Δ time units does not affect stability of the system. We provide more implementation details below.

During interval Δ, the router keeps a local variable S, which tracks the total amount of data that has arrived into the queue (counting any dropped packets as well) since the beginning of the interval. Specifically, for each incoming packet k from flow i, the router increments S by the size of the packet: $S = S + s_i(k)$. In addition, the router examines whether its locally recorded estimate $\tilde{p}$ of packet loss (which was calculated in the previous interval Δ) is larger than the one carried in the packet. If so, the router overrides the corresponding entries in the packet and places its own router ID, packet loss, and sequence number into the header. In this manner, after traversing the whole path, each packet records information from the most congested link.[3]

At the end of interval Δ, the router approximates the combined arriving rate $X(n) = \sum_{i=1}^{N} x_i(n - D_i^{\rightarrow})$ by averaging S over time Δ:

$$\tilde{X} = \frac{S}{\Delta}. \tag{71}$$

Based on this information, the router computes an estimate of packet loss $p(n)$ using

$$\tilde{p} = (\tilde{X} - C)/\tilde{X} \tag{72}$$

where C is the capacity of the outgoing link known to the router (these functions are performed on a per-queue basis).

After computing $\tilde{p}$, the router increments its packet-loss sequence number (i.e., $\mathrm{seq} = \mathrm{seq} + 1$) and resets variable S to zero. Newly computed values seq and $\tilde{p}$ are then inserted into qualified packets arriving during the next interval Δ and are subsequently fed back by the receiver to the sender. The latter adjusts its sending rate as we discuss in the next section.

C. The User

MKC employs the primal algorithm (17)–(18) at the end users who adjust their sending rates based on the packet loss generated by the most congested resources of their paths. However, to properly implement MKC, we need to address the following issues.

First, most existing congestion control algorithms are window-based, while MKC is a rate-based method. This means that, instead of sending out a window of packets at once, each MKC user i needs to properly pace its outgoing packets and maintain its sending rate at a target value $x_i(n)$. We implement this mechanism by explicitly calculating the interpacket interval $\delta_i(k)$ of each packet k:

$$\delta_i(k) = \frac{s_i(k)}{x_i(n)} \tag{73}$$

where $s_i(k)$ is the size of packet k of user i.

Second, notice that ACKs carrying feedback information continuously arrive at the end user and, for the most part, contain duplicate feedback (assuming Δ is sufficiently large). To prevent the user from responding to redundant or sometimes obsolete feedback caused by reordering, each packet carries a sequence number seq, which is modified by the bottleneck router and is echoed by the receiver to the sender. At the same time, each end user i maintains a local variable seq_i, which records the largest value of seq observed by the user so far. Thus, for each incoming ACK with sequence seq, the user responds to it only when $seq > seq_i$. This allows MKC senders to pace their control actions such that their rate adjustments and the router's feedback occur on the same timescale.

[3]Note that multipath routing is clearly a problem for this algorithm; however, *all* existing AQM congestion control methods fail when packets are routed in parallel over several paths.

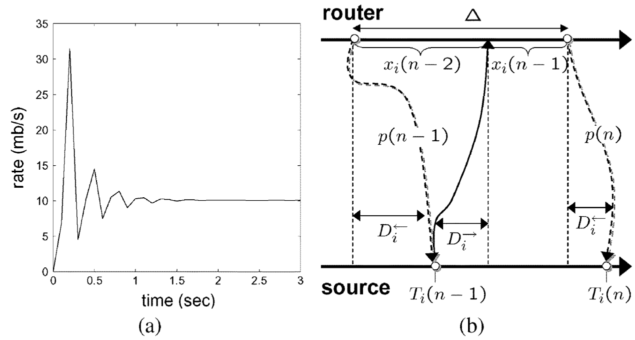


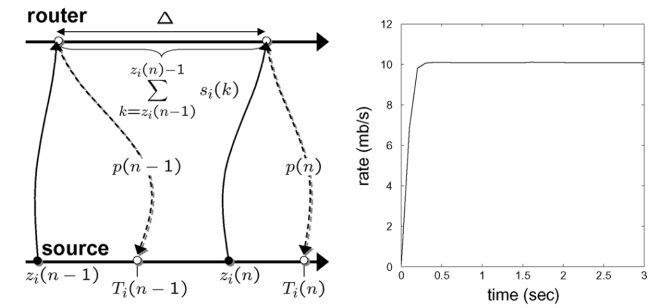
Fig. 6. Naive EMKC implementation: (a) one ns2 flow (α = 100 kb/s, $\beta = 0.9$, and Δ = 50 ms) passes through a bottleneck link of capacity 10 mb/s; (b) inconsistent feedback and reference rate.

Third, recall from (17) and (18) that MKC requires both the delayed feedback $\eta_i(n)$ and the delayed reference rate $x_i(n - D_i)$ when deciding the next sending rate. Thus, the next problem to address is how to correctly implement the control (17). We develop two strategies for this problem below.

1) Naive Implementation: One straightforward option is to directly follow (17) based on the rate that was in effect exactly D_i time units earlier. Since round-trip delays fluctuate, the most reliable way to determine $x_i(n - D_i)$ is to carry this information in the usr field of each packet (see Fig. 5). When the receiver echoes the router fields to the sender, it also copies the usr field into the acknowledgment. We show the performance of this strategy via ns2 simulations in Fig. 6(a), in which a single MKC flow passes through a bottleneck link of capacity 10 mb/s. We set α to 100 kb/s, β to 0.9, packet size to 200 bytes, and router sampling interval Δ to 50 ms. As seen from Fig. 6(a), the sending rate converges to its stationary point in less than 2 s and does not exhibit oscillations in the steady state; however, the flow exhibits transient oscillations and overshoots C by over 200% in the first quarter of a second. Although this transient behavior does not affect stability of the system, it is greatly undesirable from the practical standpoint.

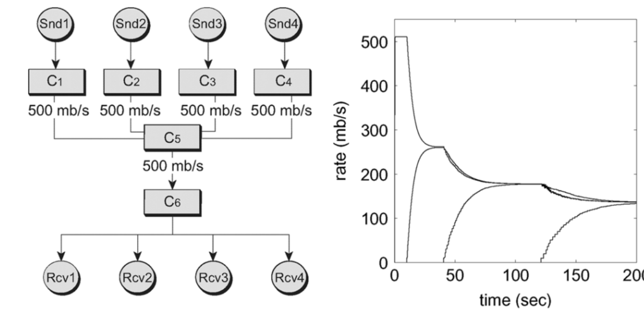
2) Proper Implementation: To remove the transient oscillations, we first need to understand how they are created. Notice from (71) that since the router calculates the packet loss based on the average incoming rate over interval Δ, it is possible that packets of *different* sending rates $x_i(n_1)$ and $x_i(n_2)$ arrive to the router during the same interval Δ. Denote by $T_i(n)$ the time when user i receives the nth nonduplicate feedback $p(n)$. Since the user responds to each feedback only once, it computes new sending rates $x_i(n)$ at time instances $T_i(n)$. To better understand the dynamics of a typical AQM control loop, consider the illustration in Fig. 6(b). In the figure, the router generates feedback $p(n-1)$ and $p(n)$ exactly Δ units apart. This feedback is randomly delayed by $D_i^{\leftarrow}$ time units and arrives to the user at instances $T_i(n-1)$ and $T_i(n)$, respectively. In response to the first feedback, the user changes its rate from $x_i(n-2)$ to $x_i(n-1)$; however, the router observes the second rate only at time $T_i(n-1) + D_i^{\rightarrow}$. At the end of the nth interval Δ, the router averages both rates $x_i(n-2)$ and $x_i(n-1)$ to produce its feedback $p(n)$ as shown in the figure.

When the control loop is completed, the user is misled to believe that feedback $p(n)$ refers to a single rate $x_i(n-1)$ and



(a) (b)

Fig. 7. Proper EMKC implementation: (a) graphical explanation of the algorithm; (b) one ns2 flow (α = 100 kb/s, $\beta = 0.9$, and Δ = 50 ms) passes through a link of capacity 10 mb/s.



(a) (b)

Fig. 8. Four EMKC (α = 10 mb/s, $\beta = 0.9$, and Δ = 100 ms) flows in the "dumb bell" topology.

forced to incorrectly compute $x_i(n)$. This inconsistency is especially pronounced in the first few control steps during which flows increase their rates exponentially and the amount of error between the actual rate and the reference rate is large.

Instead of changing the router, we modify the end users to become more sophisticated in their processing of network feedback. The key is to allow end users to accurately estimate their own contribution to $\tilde{X}$ in (71) and determine their *average* rates seen by the router during interval Δ. For each outgoing packet k, MKC sender i places the packet's sequence number k in the usr field and records in local memory the size of the packet $s_i(k)$ and its sequence number k. Upon arrival of the nth nonduplicate feedback at time $T_i(n)$, the end flow extracts the usr field from the acknowledgment and records its value in variable $z_i(n)$, which is the sequence number of the packet that generated feedback $p(n)$. To compute the new rate $x_i(n)$, the user calculates the amount of data that it has transmitted between packets $z_i(n-1)$ and $z_i(n)-1$ and normalizes the sum by Δ, which is exactly the average rate used by the router in generation of $p(n)$.

To visualize this description, consider Fig. 7(a), in which the end flow is about to decide its sending rate $x_i(n)$ at time $T_i(n)$. Notice in the figure that feedback $p(n)$ is based on all packets of flow i with sequence numbers between $z_i(n-1)$ and $z_i(n)-1$. Through the use of $z_i(n)$, we obtain a projection of the time interval used by the router in its computation of $p(n)$ onto the sequence-number axis of the end user.[4] Given the above discussion, the user computes its average rate as

$$\bar{x}_i(n) = \frac{1}{\Delta} \sum_{k=z_i(n-1)}^{z_i(n)-1} s_i(k) \tag{74}$$

and utilizes it in its control equation

$$x_i(n) = \bar{x}_i(n) + \alpha - \beta \eta_i(n) \bar{x}_i(n). \tag{75}$$

Next, we turn our attention to the ns2 simulation in Fig. 7(b) and examine the performance of this strategy with a single flow.

[4]Note that this approach is robust to random delays but may be impeded by severe packet loss at the router.

The figure shows that (74) and (75) successfully eliminate transient oscillations and offer fast, monotonic convergence to the steady state.

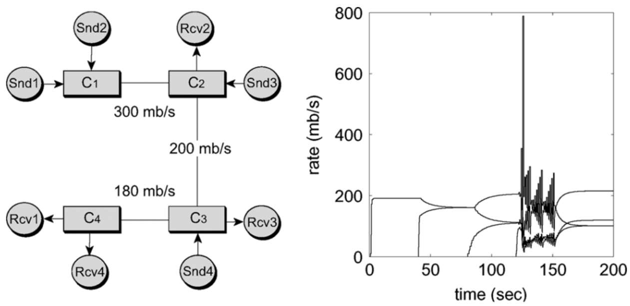
Our next example shows the performance of the new implementation (74)–(75) with multiple flows. The simulation topology of this example is illustrated in Fig. 8(a): four EMKC flows access a common bottleneck link of capacity 500 mb/s. The round-trip propagation delays of the four flows are, respectively, 10, 100, 500, and 1000 ms. As Fig. 8(b) shows, flow x_1 starts with an initial rate 100 kb/s and reaches link utilization in less than 1 s. When flow x_2 joins at time 10 s, a flow rate of x_1 is driven down toward the new stationary rate 261.1 mb/s, and 99% fairness is achieved in 25 s. This behavior repeats as flows x_3 and x_4 start, respectively, at time 40 and 120 s, and the system quickly restabilizes in the new equilibrium without any transient oscillations.

D. Multi-Link Simulations

After demonstrating EMKC's single-link performance, we proceed to examine the multilink topology illustrated in Fig. 9(a). In this topology, capacities of links C_1–C_2, C_2–C_3, and C_3–C_4 are respectively 300, 200, and 180 mb/s, the corresponding round-trip propagation delays are 10, 500, and 100 ms, and the sampling intervals Δ are 100, 100, and 99 ms. In the network, there are three short flows x_2–x_4, respectively, utilizing links C_1–C_2, C_2–C_3, and C_3–C_4 and one long flow x_1 passing through all three links.

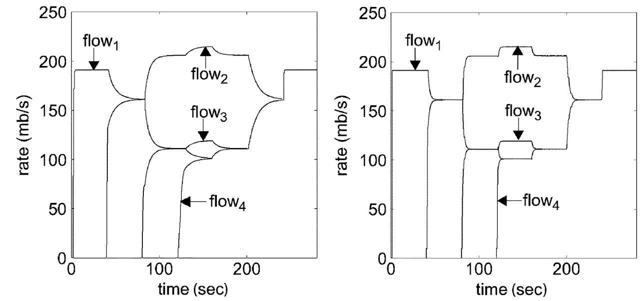
The simulation result of EMKC employing implementation (74)–(75) is plotted in Fig. 9(b). Flow x_1 starts first and reaches the utilization of the bottleneck link C_3–C_4 in 2 s. As flow x_2 joins at time 40 s, the bottleneck of x_1 switches to link C_1–C_2 and both x_1 and x_2 converge to fairness. Similarly, when x_3 starts at time 80 s, link C_2–C_3 becomes the new bottleneck of x_1. As a consequence, x_1 and x_3 converge toward the new fair rate and x_2 climbs up and collects the residual bandwidth on link C_1–C_2. The individual sending rates are smooth during the first 120 s. However, after flow x_4 joins, the system suffers sustained oscillations during the period from time 124 through 151 s. The oscillation is especially severe for x_1, whose sending rate reaches as high as 788.5 mb/s at time 125.7 s, overshooting the bottleneck C_3–C_4 by over 300%.

We next investigate the underlying reason for this oscillation. Observe that as the sending rate of x_4 increases, the bottleneck



(a) (b)

Fig. 9. (a) "Parking lot" topology; (b) Naive implementation of EMKC ($\alpha =$ 10 mb/s and $\beta = 0.9$) in the "parking lot" topology.



(a) (b)

Fig. 10. Proper implementation of EMKC ($\alpha = 10$ mb/s, $\beta = 0.9$, and $\delta = 0.01$) in the "parking lot" topology. (a) Constant delay; (b) random delay.

of flow x_1 switches from link C_2–C_3 to C_3–C_4. Since it is possible for this switching to occur in the middle of the bottleneck router's sampling interval, the computed packet loss could be inconsistent with the end user's reference rate. This results in fluctuations in the sending rate and is the primary reason for the "spike" shown in Fig. 9(b). Moreover, this situation could be exacerbated when multiple resources with close capacities (e.g., 180 and 200 mb/s in this case) exist in the path of a certain user, since fluctuating input rate at the routers will cause fluctuating packet loss, which could eventually lead to oscillations of the bottleneck link and aggravate the rate oscillations of the end users. This explains the oscillations after the spike.

We emphasize that these problems do not indicate instability of EMKC, but arise as the result of discretized implementation of the theoretical model given by (32) and (48). To properly deal with multibottleneck networks, we develop several strategies to manage bottleneck switching. First, we force the end user to delay its response to the ACK for one RTT once a bottleneck switch is detected. By doing this, the packet loss carried in the next nonduplicate ACK will be consistent with the reference rate $\bar{x}_i(n)$ computed by the user. Second, we damp the bottleneck oscillations resulting from multiple routers with close capacities by introducing a threshold value δ such that the end user authorizes a bottleneck switch if only if the difference between the old packet loss and the packet loss carried in the ACK is greater than δ.

To examine the effectiveness of this mechanism, we redo the simulation in Fig. 9(b) using this new algorithm and plot the result in Fig. 10(a). As seen in the figure, this implementation removes the oscillations that originally occurred when x_4 joined the system. Starting from time 160 s, flows x_4, x_3, and x_2 terminate with a 40-s delay, and there is no oscillation in both the transient phase and the steady state.

We next incorporate randomness into the feedback delay of individual flows and test EMKC in settings with highly variable delays. To implement time-varying delay, we maintain a local queue at the receiving end of each flow and force the ACKs to pass through this queue before being echoed to the sender. For every m successfully transmitted acknowledgments, the system delays the head packet in the queue by d seconds and the other packets by 10 μs. Here, time-varying variables d and m are uniformly distributed in [0.5, 1] and [500000, 1000000], respectively. All packets between the d-second delay spikes are drained at the wire speed of the return path, which ensures that the queue is completely emptied before the next spike is generated. We preserve the topology in Fig. 9(a) except that the round-trip propagation delay of each flow is fixed to be 10 ms such that the effect of random delay is more evident. The simulation result is depicted in Fig. 10(b), in which the system exhibits delay-independent asymptotic stability, fast convergence to the stationary point, and smooth transitions between the neighboring states.

VII. Conclusion

This paper investigated the properties of Internet congestion controls under non-negligible directional feedback delays. We focused on the class of control methods with radial Jacobians and showed that all such systems are stable under heterogeneous delays. To construct a practical congestion control system with a radial (symmetric in particular) Jacobian, we made two changes to the classic discrete Kelly control and created a max–min version we call MKC. Combining the latter with a negative packet-loss feedback, we developed a new controller EMKC and showed in theory and simulations that it offers smooth sending rate and fast convergence to efficiency. Furthermore, we demonstrated that EMKC's convergence rate to fairness is exponential when the network provider scales the number of flows N as $\Theta(C)$ and linear otherwise. From the implementation standpoint, EMKC places very little burden on routers, requires only two local variables per queue and one addition per arriving packet, and allows for an easy implementation both in end-to-end environments and under AQM support. Our future work involves improvement of the convergence speed to fairness and design of pricing schemes for EMKC to achieve loss-free performance regardless of the number of flows N.

References

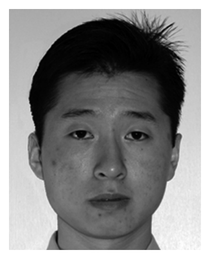
[1] R. Bronson, *Schaum's Outline of Theory and Problems of Matrix Operations*. New York: McGraw-Hill, 1988.

[2] D.-M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Comput. Netw. ISDN Syst.*, vol. 17, no. 1, pp. 1–14, Jun. 1989.

[3] M. Dai and D. Loguinov, "Analysis of rate-distortion functions and congestion control in scalable internet video streaming," in *Proc. ACM NOSSDAV*, Jun. 2003, pp. 60–69.

[4] S. Deb and R. Srikant, "Global stability of congestion controllers for the internet," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 1055–1060, Jun. 2003.

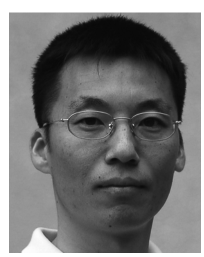
[5] S. Floyd, "High-speed TCP for large congestion windows," IETF RFC 3649, Dec. 2003.
[6] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proc. ACM SIGCOMM*, Aug. 2000, pp. 43–56.
[7] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, Jan. 1993.
[8] R. A. Horn and C. R. Johnson*, Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1990.
[9] C. Jin, D. Wei, and S. H. Low, "FAST TCP: Motivation, architecture, algorithms, performance," in *Proc. IEEE INFOCOM*, Mar. 2004, pp. 2490–2501.
[10] R. Johari and D. K. H. Tan, "End-to-End congestion control for the internet: Delays and stability," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 818–832, Dec. 2001.
[11] K. Kar, S. Sarkar, and L. Tassiulas, "A simple rate control algorithm for maximizing total user utility," in *Proc. IEEE INFOCOM*, Apr. 2001, pp. 133–141.
[12] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth delay product networks," in *Proc. ACM SIGCOMM*, Aug. 2002, pp. 89–102.
[13] W. G. Kelley and A. C. Peterson*, Difference Equations*. New York: Harcourt/Academic, 2001.
[14] F. P. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Telecommun.*, vol. 8, no. 1, pp. 33–37, Jan. 1997.
[15] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, no. 3, pp. 237–252, Mar. 1998.
[16] T. Kelly, "Scalable TCP: Improving performance in high-speed wide area networks," in *Proc. 1st Int. Workshop Protocols for Fast Long-Distance Networks*, Feb. 2003.
[17] S. Kunniyur and R. Srikant, "Analysis and design of an Adaptive Virtual Queue (AVQ) algorithm for active queue management," in *Proc. ACM SIGCOMM*, Aug. 2001.
[18] S. Kunniyur and R. Srikant, "A time-scale decomposition approach to adaptive explicit congestion notification (ECN) marking," *IEEE Trans. Autom. Control*, vol. 47, no. 6, pp. 882–894, Jun. 2002.
[19] S. Kunniyur and R. Srikant, "End-to-End congestion control schemes: Utility functions, random losses and ECN marks," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 689–702, Oct. 2003.
[20] S. Kunniyur and R. Srikant, "Stable, scalable, fair congestion control and AQM schemes that achieve high utilization in the internet," *IEEE Trans. Autom. Control*, vol. 48, no. 11, pp. 2024–2029, Nov. 2003.
[21] D. Loguinov and H. Radha, "End-to-End rate-based congestion control: Convergence properties and salability analysis," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 564–577, Aug. 2003.
[22] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Trans. Netw.*, vol. 11, no. 4, pp. 525–536, Aug. 2003.
[23] S. H. Low and D. E. Lapsley, "Optimization flow control I: Basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 861–874, Dec. 1999.
[24] L. Massoulié, "Stability of distributed congestion control with heterogeneous feedback delays," *IEEE/ACM Trans. Netw.*, vol. 47, no. 6, pp. 895–902, Jun. 2002.
[25] T. Mori, N. Fukuma, and M. Kuwahara, "Delay-independent stability criteria for discrete-delay systems," *IEEE Trans. Autom. Control*, vol. 27, no. 4, pp. 964–966, 1982.
[26] F. Paganini, J. Doyle, and S. H. Low*, A Control Theoretical Look at Internet Congestion Control*, ser. Multidisciplinary Research in Control: The Mohammed Dahleh Symposium 2002. : Springer-Verlag, 2003.
[27] G. Vinnicombe, "On the stability of end-to-end congestion control for the Internet," Cambridge Univ., Cambridge, U.K., Tech. Rep. CUED/F-INFENG/TR.398, Dec. 2000.
[28] G. Vinnicombe, "On the stability of networks operating TCP-like protocols," in *Proc. IFAC*, Aug. 2002.
[29] G. Vinnicombe, "Robust congestion control for the Internet," Cambridge Univ., Cambridge, U.K., Tech. Rep., 2002.
[30] B. P. Wydrowski, L. L. H. Andrew, and I. M. Y. Mareels, "MaxNet: Faster flow control convergence," *Netw.*, vol. 3042, pp. 588–599, May 2004.
[31] B. P. Wydrowski and M. Zukerman, "MaxNet: A congestion control architecture for maxmin fairness," *IEEE Commun. Lett.*, vol. 6, no. 11, pp. 588–599, Nov. 2002.
[32] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control for fast, long distance networks," in *Proc. IEEE INFOCOM*, Mar. 2004, pp. 2514–2524.
[33] Y. R. Yang and S. S. Lam, "General AIMD congestion control," in *Proc. IEEE ICNP*, Nov. 2000, pp. 187–198.
[34] L. Ying, G. E. Dullerud, and R. Srikant, "Global stability of internet congestion control with heterogeneous delays," in *Proc. Amer. Control Conf.*, Jun. 2004.
[35] Y. Zhang, S.-R. Kang, and D. Loguinov, "Delayed stability and performance of distributed congestion control," in *Proc. ACM SIGCOMM*, Aug. 2004, pp. 307–318.
[36] Y. Zhang, D. Leonard, and D. Loguinov, "JetMax: Scalable maxmin congestion control for high-speed heterogeneous networks," in *Proc. IEEE INFOCOM*, Apr. 2006.
[37] Y. Zhang and D. Loguinov, "Local and global stability of symmetric heterogeneously-delayed control systems," in *Proc. IEEE CDC*, Dec. 2004, pp. 5004–5009.



Yueping Zhang (S'03) received the B.S. degree in computer science from the Beijing University of Aeronautics and Astronautics, Beijing, China, in 2001. He is currently working towards the Ph.D. degree in computer engineering at Texas A&M University, College Station.

His research interests include congestion control, delayed stability analysis, active queue management (AQM), and peer-to-peer networks.

Mr. Zhang has been a student member of the Association for Computing Machinery (ACM) since 2003.



Seong-Ryong Kang (S'04) received the B.S. degree in electrical engineering from Kyungpook National University, Daegu, Korea, in 1993 and the M.S. degree in electrical engineering from Texas A&M University, College Station, in 2000. He is currently working towards the Ph.D. degree in the Department of Computer Science at Texas A&M University.

From 1993 to 1997, he worked as a patent engineer/consultant for First IPS, Korea. His research interests include media streaming, congestion control, and bandwidth estimation.

Mr. Kang has been a student member of the Association for Computing Machinery (ACM) since 2005.

Dmitri Loguinov (S'99–M'03) received the B.S. degree (with honors) in computer science from Moscow State University, Moscow, Russia, in 1995 and the Ph.D. degree in computer science from the City University of New York, New York, in 2002.

Since 2002, he has been an Assistant Professor of computer science with Texas A&M University, College Station. His research interests include peer-to-peer networks, Internet video streaming, congestion control, image and video coding, Internet traffic measurement, and modeling.